Data Splash!
An Educational Game about Machine Learning

Andrew Gray

445348

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Master of Science

Department of Computer Science

Swansea University

September 20th, 2020

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed	....	(candidate)
Date	....	

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed	....	(candidate)
Date	....	

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed	....	(candidate)
Date	....	

I would like to dedicate this work to the Hypnotoad.
All glory to the Hypnotoad.

Abstract

As part of our Masters of Science accreditation, we must complete a research thesis. In has been decided upon to create an educational game centred around Machine Learning (ML), due to the authors desire to gain a deeper understanding of ML and their previous experiences as being a secondary school teacher. We have proposed a game that allows the player to interact with different key ML algorithms and models while providing mediums to help educate and teach the players the understanding of the ML and provide knowledge on how they operate. We will achieve this by creating learning research to accompany the game, as well as links to relevant scientific research to get a deeper understanding. While at the centre of it all, having a fun and engaging game, that uses ML to teach about ML. Through using Python, Pygame and industry-standard accepted libraries and packages, like Tensorflow and Sci-kit Learn. The game will provide key gameplay features that users would expect of games, which will be achieved by using fundamental gamification techniques, to create a fun and engaging game that allows the user to interact directly with the ML models.

Acknowledgements

First, I would like to thank my partner and my daughter for allowing me to pursue furthering my education. I would also like to thank my mother for all the support she has given through this adventure. Secondly, I would like to thank all the lectures that have taught me. I have much appreciated your knowledge and wisdom that you have passed on. Finally, I would like to say a massive thank you to my supervisor, Dr Michael Edwards, and my tutor, Dr Anton Setzer. An additional thanks I would like to give it to Thomas Tasioulis and Amal Abdulkader for befriending me during this MSc and all the help and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As part of our Masters of Science accreditation, we must complete a research thesis. In has been decided upon to create an educational game centred around Machine Learning (ML), due to the authors desire to gain a deeper understanding of ML and their previous experiences as being a secondary school teacher.

1.1 Overview of the Problem

Machine learning gets perceived as a black box, a form of computer wizardry, where unknown algorithms do some magical unknown thing. Some misconceptions people have about Artificial Intelligence (AI) and ML is that 'AI does not need humans' and that 'AI is dangerous' [2]. Other misconceptions about AI and ML is that they have both very new and based around a human's brain, but AI and ML is something that has been around for a long time, and it is nowhere near the same, even at a fundamental level. Another big misconception is that AI is smarter than humans and that the ML robots will come and destroy the humans wiping out humanity. However, while AI can be better at performing specific tasks, they are not genetically more intelligent than humans. AI will only do what it gets told to do, nothing more [2].

On the other hand, instead of fearing AI and ML, there is a lot of this that it is currently doing to help humankind and make things safer. For example, the RAC, one of the UK's largest motoring organisations, aims to try and detect low-speed car crashes. They do this by developing an onboard crash sensing system that uses advanced machine learning algorithms to detect low-speed collisions and distinguish these events from more common driving events, such as driving over speed bumps or potholes. Independent tests showed the RAC system to be 92% accurate in detecting test crashes, allowing them to be able to enable rapid response to roadside incidents [3].

1.2 Overview of the Solution

The overall aim of the proposed solution is to create a fun, educating game about ML. The players will be, at the core of the solution, playing a game that interacts with different ML models. The player(s) will be manipulating the game board and data points to affect the decision boundary, or to figure out where the decision boundary or centre of the cluster is. The solution will get created by using Pygame and will have many different algorithms in the background, doing the main game mechanics, through using libraries like SKLearn [4] and Tensorflow [5].

1.3 Motivations

1.3.1 Aims & Objectives

The proposed solution aims to create a fully interactive game that users will find fun and engaging, while still providing a level of education to teach the players what the different algorithms are and how they work. From the experience of being a teacher, learning has the most impact when the learner gets able to fully interact with the learning subject and see it work first hand, rather than just being told about it. The aim by creating this educational game is to help inform people what ML is and what it does, aiming to demystify the myths and misconceptions around ML.

1.4 Contributions

The main contributions of this work can be seen as follows:

- **A written thesis**

  A document that is indended to partner and explain aspects of the final application, explain decisions made and explain the research discovered to influence decisions.

- **An education game application about machine learning**

  An application created that allows the player or user to interact with, and manipulate, different machine learning models. Additional content, in the form of a website, has additionally been provided and hosted online. This supplementary content is to help with the teaching and learning of the main ML concepts used within the application.

1.5 Thesis Overview

Chapter 2

Background & Literature Review

2.1 Introduction to Edu-Games

2.1.1 Gamification

Presented to us has been a task to research within a subtopic of Human-Computer Interaction (HCI). The subtopic we chose was gamification and gamification within education. Researching into gamification within educations has been influenced by the author's previous experience of being a teacher and working within schools. We wanted to find out what the context of gamification is, and how it can be used within education, to take aspects of teaching and learning that can get brought into the 21st Century. To make aspects of education more accessible to students in a manner that they are more accustomed to in their everyday lives.

Gamification, a term first coined in 2002, is an HCI technique used to add a game layer to traditional non-game like situations. The gamification aims to create extrinsic motivators for a person to be encouraged to do particular actions. Each action, upon completion, will have a little reward which, upon doing so, will release dopamine into the brain. The release of dopamine creates a good feeling within the participant's mind, which in turn is encouraging them to do it again. These rewards can be in the form of badges, achievements or progress bars, to name a few.

The term gamification first appeared in the context of software design in 2008 [6], but the term only started to get more widespread recognition within 2010. However, the term "gamification" was first coined by Nick Pelling in 2002 [7]. Its initial aim was to incorporate the social and reward features of games into the software. Gamification started to gain much attention, so much so that it got described by a venture capitalist as one of the most promising areas of gaming [8]. Gamification is now known as a powerful tool for engagement, which has, since its initial conception, now become a standard feature within software development [7]. Researchers consider gamification to be the progression of earlier work that focuses on adopting game-design elements to non-game situations and contexts. Research in the HCI field, in regards to apps that use game-driven features for motivation and also in interface design, suggest that there is a connection between Soviet concepts of socialist competition and the American management trend of "fun at work" [8].

Jane McGonigal, in 2010, delivered a groundbreaking TED Talk titled, "Gaming Can Make a Better World' [9]. This talk gets reflected as the defining moment in the history of gamification. Within the talk, she foretells a game based utopia. Where she states that "When I look forward to the next decade, I know two things for sure: that we can make any future we can imagine, and we can play any games we want, so I say: Let the world-changing games begin [9]." Hindsight tells she was correct, as, from 2011, gamification starts to pick up steam. At a Computer-Human Interaction (CHI) conference, a workshop titled "Gamification: Using Game Design Elements in Non-Gaming Contexts [10]", which generated the Gamification Research Network (GRN) [11], in the year 2011. Through the years 2012 to 2016, gamification continues to grow. Even so, that gamification goes viral without people knowing through a game called Pokémon Go. Pokémon Go is one of the most successful applications of gamification with over 800 million downloads. People who would usually turn their nose up at badge collecting were out patrolling the streets searching for rare Pokémon. Pokémon Go is one of the most successful apps of all time. It even broke records [7, 12]. It could be said thanks to Pokémon Go, that gamification is now everywhere.

Many established technology and other companies, including SAP AG, Microsoft, IBM, SAP, LiveOps, Deloitte, and other companies have started using gamification in various applications and processes [13].

The increased popularity in gamification, within some contexts, has had led to many legal restrictions be placed upon it, especially when linked to the Internet of Things (IoT) features. However, this mainly refers to the use of virtual currencies and assets, as well as data privacy, data protection and labour laws. These laws are due to its nature of being a data mining systems that spread information online, known as data aggregator [11, 14].

2.1.2 Gamification in Education

The gamification of learning is an educational approach to motivate students to learn by using game elements in a learning environment [15]. Gamification in education is very much the same thing as gamification in general, but with more of a focus on learning. However, gamification in learning has two main views within HCI academia. One side categories gamification of learning as learning that has game-like features, but only when the learning is happening in a non-game context, like a classroom. This version would involve a range of components that get presented in a system, or game layer, which aims to happen alongside the learning in a conventional classroom. At the same time, the other half includes games that have been designed to induced learning within them [15].

Gamification, within an educational or a learning situation, has multiple advantages. It is not just about trying to improve attendance with incentives by reaching a particular score, or extra rewards for completing specific tasks within a lesson. It can aid in cognitive development in adolescents, which can increase levels of engagement and can aid with accessibility within the classroom [16]. Games that get produced for enhancing cognitive development are known as "rain games" [16]. These popular games typically are focused around a series of questions and problems for the player to solve or answer. These games develop the rate the player can sustain information and increase the brain's ability to process information. The levels of the engagement of the students' increases, when gamification has been used, within a classroom. A study performed by scientists aimed to measure the students' levels of engagement in a classroom where gamification elements are applied [17]. They assigned a point system to multiple daily activities. Every student had a measurement of the perceived level of engagement. The finding showed that the game like setting was supporting the learning within the classroom and increased productivity. Therefore, by increasing engagement levels, it also means it helps students be able to access the content of the lesson, that is or needs to be delivered better.

Even though gamification can aid teaching students of all needs, a study conducted on students who had autism using video games showed that this training package was powerful in teaching content that was age-appropriate [18]. However, gamification of learning is not something just for the classroom; its an excellent tool for learning outside the classroom. Games like Spore create a deeper understanding of life and evolution as the game simulates a world where the player's character will evolve, adapting to their surroundings through reproduction. Another game by the same creator, Will Wright, Sim City aims to teach the player key skills like [19]: Supply and demand; Budgeting; Urban planning; Managing the environment; Understanding utilities and services like transport systems and public services; Reading and maths skills.

Gamification of learning has excellent potential benefits. The benefits involve [15]: Allowing students to have ownership of their learning, as well as giving opportunities for the learner to gain a sense of their' own identity" through alternative role-playing selves. The freedom, without any negative repercussions, to fail and keep on trying again. The ability to increase fun and joy while learning. The opportunity for tasks to be differentiated. Making the learning visible and providing opportunities to inspire intrinsic motivators for learning. Also, the ability to aiding in motivating students with low levels of motivation.

Gamification in Science

Although science concepts still use more conventional styles of gamification. Science concepts will often use a type of gamification game that has a primary purpose, which is other than just for pure fun, called a 'serious' or 'applied' game. These types of games get utilised by industries like scientific exploration, education, health care, defence, emergency management, city planning, engineering and politics [20]. Although not all do, serious games tend to share aspects closely tied with simulational games. However, all serious games still have other gamification features included (see fig: **??**).

Nonetheless, in regards to the field of science, serious games' role is to include crucial activities for scientists. These include outreach, teaching and research. With serious games on the increase, an emerging sub-genre is called citizen science games (CSGs) [21]. CSGs enables the user to produce as well as, or instead, analyse data for scientific use. Some examples of CSGs are GalaxyZoo, Foldit and HiRE-RNA [22, 23]

Studies suggest that there are ten main rules for serious games to follow. These are [21]:

1. Define a serious goal - we must first define the purpose of the game at the beginning of its development. Is its purpose for science, outreach, teaching or a combination of all three?

2. Get the balance between entertainment and serious tasks - the game design should be implemented as a function of the objectives of the game. Therefore equilibrium and compromise need to be found between scientific accuracy and player accessibility.

3. Allow the player to interact with the scientific data - players interest increases if they can interact with the science data, enriching the learning experience. The ability for players to generate data also creates another perspective for the player, increasing interaction.

4. Promote onboarding and engagement - Expectations of players are varied. Therefore the reward system needs to be versatile. Ideally, the entry-level should be low and the difficulty altered to each player.

5. Manage Information Flow - How the information to the play gets received will impact their behaviour, either positively or negatively. So if the focusing is on the outcome, this could influence the results.

6. Provide an appropriate narrative - This is important for all games, but also crucial for serious games. The narrative should give the player context to the game, allowing them to know what to do.

7. Adapt the level design - Depending on the objective, variation on level designs needs implementing. These can include duration, tasks and difficulty.

8. Develop good graphics that are not just pleasing on the eye - High-quality graphics increase the player's immersion into the game.

9. Use all modalities, especially sound - Using just a visual channel can overload the player. Therefore it is vital to take the load of the player's vision and use several different channels — for example, sound.

10. Iteratively assess what works and what does not - However, it is vital to take into account three different perspectives for serious games. The developer, the player and the scientist as they all have different views on what they believe the game needs adapting based on their desires.

2.1.3 Example Applications

2.2 Machine Learning

2.2.1 Machine Learning Fundamentals

2.2.2 Supervised vs Unsupervised Learning

Machine learning uses two types of techniques: supervised learning, which trains a model on known input and output data so that it can predict future outputs, and unsupervised learning, which finds hidden patterns or intrinsic structures in input data [24].

Supervised:
Supervised machine learning aims to build a model that makes predictions based on evidence in the presence of uncertainty. The supervised learning algorithm takes the insights it has gained, from a known set of input data and known responses to the data (output), also known as labels, and trains a model to generate reasonable predictions for the response to new data [3, 24].

Supervised learning uses classification and regression techniques to develop predictive models. Classification is a technique that predicts discrete responses. The classification aims to classify the inputted data into different categories [3]. Some examples of this type of technique are deciding if an email is spam or not, or deciding if a patient has a benign or cancerous tumour. These types of applications include credit scoring, medical imaging and speech recognition.

On the other hand, regression techniques try to predict continuous responses [24]. An excellent example of this is to check for changes in the temperature or checking the power demands fluctuations. This kind of applications would get used for trading and forecasting electricity load [3].

Unsupervised Learning:
Unsupervised learning aims to find hidden patterns or intrinsic structures in the data [24]. In the same regards as supervised learning, unsupervised learning aims to gain insights from the data. However, whereas supervised learning has the output labels for the provided dataset, unsupervised does not, it aims to explore the data to find patterns or groupings in the data [3].

Examples of clustering applications include gene sequence analysis, market research, and object recognition [3].

2.2.3 Methods

2.2.3.1 K-Means

2.2.3.2 Gaussian Mixture Model

2.2.3.3 Neural Network

Neural Networks (NN) or also known Artificial Neural Networks (ANN), are at the very core of Deep Learning (DL). An ANN is an ML model inspired by networks of biological neurons found in our brains. However, ANN has become very different from their biological cousins that they took inspiration. ANN is what powers Google's image classifications, Apple's Siri and YouTube's video recommendation, as well as learning to beat the worlds best player at the game called Go, named 'DeepMind's Alpha Go' [24].

2.2.3.4 Linear Regression

Linear regression is a model that aims to fit a line of best fit to the data provided. In order to achieve the best fit, the algorithm chooses the best overall score from the 'Root Mean Square Error' (RMSE). Therefore, to train a linear regression model, we need to find the value of 0 that minimises the RMSE [24].

2.2.3.5 Logistic Regression

Logistic regression is an algorithm that gets used for classification. Logistic regression gets used to estimate the probability that an instance belongs to a particular class. If the estimated probability is greater than 50%, the model predicts that the instance belongs to that class. If the model has predicted that the instance belongs to that class, also known as the positive class, it gets a '1' label [**?**].

2.2.3.6 SVM

Support Vector Machine (SVM) is another powerful and versatile ML model. The model is capable of performing linear or nonlinear classification, regression and even outlier detection. SVM is one of the most popular ML models and are best suited for small to medium-sized datasets. SVM aims to separate the data categories by using a decision boundary, with the largest margin between them. The algorithm is known as the 'large margin classification' [24].

2.2.3.7 Linear Discriminant Analysis

2.2.3.8 PCA

2.2.3.9 kNN

2.2.4 Machine Learning in Education

2.2.4.1 Classical Approaches

2.2.4.2 Machine Learning Edu-Games

2.3 Proposed Solution

The overall aim of the proposed solution is to create a fun, educating game about ML. The players will be, at the core of the solution, playing a game that interacts with different ML models. The player(s) will be manipulating the game board and data points to affect the decision boundary, or to figure out where the decision boundary or centre of the cluster is. The solution will get created by using Pygame and will have many different algorithms in the background, doing the main game mechanics, through using libraries like SKLearn [4] and Tensorflow [5].

2.4 Summary and Overview of Proposed Solution

Chapter 3

Methodology

3.1 Overview of Application

The application has three main segments. These segments are a game area, a learning area and an exploring area. Each area's intension is to help support the user learning and understanding of the different machine learning models using a blend of exploration, fun and interactivity as well as a more traditional teaching a learning style of quizzes and learning reading material. Although each segment has its core task, together they help give the user a rounded learning experience while creating gamification incentives to come back and use the application some more.

3.1.1 Design

With the application being about the user interacting with data, and placing (splashing) the data points around a game board, we decided upon the title "Data Splash". With the title 'Data Splash" agreed upon, a beach and sea themed colour pallet got chosen. The pallets colours contained blue, yellow, turquoise and orange. However, additional colours got used to aid the colour pallet selected, and these colours involved grey and red.

All the screens had a similar layout, with a title banner image at the top, the content in the middle and the buttons in the bottom general area. The only screens that are different are the main menu screen and the coming soon splash screen. The main menu did follow a similar structure, but the main content was the buttons, which get presented in a horizontal stage manner (see fig: ??).

Whenever a model or educational content got made available to the player, this content was the main focus to the screen. Therefore always making sure that the user's attention was on interacting with the model or learning about them. Unless like in the Free Play area, both learning and model interaction was available, equal weighting occurred given to allow focus on interacting and learning about the machine learning models.

3.2 Overview of Specific Game Components

3.2.1 Game Arena

The 'Game Zone' was the critical area that intended to use game mechanics, and gamification, to help drive the learning of the different machine learning models. The game zone is an area that allows one on one (player vs player) game action. The game gets conducted over three rounds, with each game round having a random model generated for the players to interact. At the time of writing this report, the available models for the game zone are Linear Regression and K-Means. A Nural Network got implemented with game mechanics, but due to time restrictions, we were unable to add them to the application in time. As the research suggested, having a competitive nature to the game creates desired external motivation for the player to learn more about the ML models, to be able to have a better chance of winning. A running score is presented to the players to let them know who won the previous round and who is currently winning. The multiple forms of game stats allow the players to have an idea of what is needed to win the game potentially through using sport like game mechanics to add the layer of progress updates continuously, to create that sense of competitiveness and external motivation. Even though there were only two models implemented fully into the game, these models offered several different gaming outcomes. For example, each model had multiple datasets which would get selected at random. In terms of K-Means, a random dataset would get generated each time or a preselected dataset, but the k value would change and be a random number. So even though the dataset was a k value of 2, the challenge would be added by not knowing what k value was given to the model for the user to predict the centroid value.

We selected K-Means and Linear Regression to be implemented first due to them both having a similar game mechanic intention. Linear Regression was using the SSE metric value as the deciding factor to determine the game's winner, while K-Means was using the model's metric value of the euclidean distance to do the same thing. The neural network uses a territory-based mechanic, and this intention was to add variety in not only the models but the game times. Challenging the players understanding of how different models work.

With having multiple models available, as well as getting the models and their datasets randomly selected each time, this allows the game to feel fresh each time and not have a set pattern of motions. Therefore, by creating a sense of game mode uncertainty or randomness will keep things fresh. Thus, ultimately making sure that a critical mechanic of gamification, which is replayability, be achieved.

There were intentions for the Learning Zone to also provide example code for the user, after specific gamification actions, for example getting full marks in the quiz, were completed as bonus rewards. The intention of this was to allow the users to not only learn about the code but also see the code, to help see the mechanics in it. However, due to time restrictions and certain gamification features not being implemented, this additional feature was not added at this point.

3.2.2 Learning Zone

The learning zones aim is for the user to do the principal amount of learning about the different machine learning models. The main content gets presented to the user by using a web browser widget. This widget would link to a multipage HTML website that holds all the content about the different models with pictures. We decided to use this combination as it allowed us to update the learning content and add new content as we went along, enabling the main functionality not get affected. Also, it avoided unnecessary long developing time, because of the updates and the new content, forcing the redesign of the game screen.

With the teaching and learning getting conducted through text and images on the webpages, we decided to add a quiz. The quiz was to allow the user to assess how much they have learnt. A useful tool used by teachers to evaluate students learning is different questioning techniques. In an attempt to allow the users to test their subject knowledge, but keeping everything in a game-like manner, a quiz was implemented. The quiz, with not only challenging the user but also offers an overall score allowing the user to know how they did and will enable a form of competition to happen and also let the use sense a way of progression by observing their performance improving.

An option available to the user is not only to quiz themself but also to have the ability to go straight to the Free Play area. When the user selected this option, the model that the user has been learning about will be preloaded into the screen, allowing them to be able to interact with it. We decided to do this as although you can learn a lot about a topic by reading about it, an effective way to truly learn about something is to be able to interact with it and see what is happening by, in essence, trying to break it in a sort of way.

3.2.3 Free Play

The free play area is where the serious gamification style gets implemented. This area intends to allow users to be able to interact with the ML models. Initialise not only the models but also set the type of data they want to have displayed and even add additional data points. By allowing the user to add extra data points to the existing data, it will update the model and, for example with Linear Regression, will show to the user, how the additional data points will affect the model's decision making.

The most interactive model's within the FP area, at current, are the models Linear Regression, K-Means, LDA and Neural Networks. However, the models GMM and SVM are also available, but with less interactivity as the others. K-Means allows the user not only to select different data sample but also select how many clusters they want the dataset to have, but also independently change the k value of clusters. We decided upon this feature to allow the user to see, knowing how many k clusters the dataset has, to see by changing the algorithms k value how is that then affected on the dataset with its outcome. The user can click within the Matplotlib widget and see what the data points prediction values are as well. We decided to use a click in the widget functionality, as we believed having the user input x and y coordinates would make the UI look cumbersome and add unneeded fiddliness for the user, trying to figure out the exact values they want. Linear regression allows the player to be able to make predictions, as well as additional data points allowing them to see how the data can be altered and manipulate and what implications that has on the models fitting. However, Linear Regression does not provide as much control for the intricacies of the model compared to K-Means, but it does offer the parameters and outputs that the model has. For example, the intercept and the coefficients to the model. Neural Networks offer slightly less control to the user compared to the other two, but more than LDA but both of them have fully interactive models. SVM and GMM, on the other hand, do not. GMM allows the user to switch its predictions on and off, but SVM just shows the models predictions. These got implemented to help with understanding the content from the learning zone, but we decided to focus on the other models due to them having more game-like features to get used in the game area.

3.2.4 Awards Zone

The achievement area has just a title, coming soon image and a button. Once the button gets pressed, it will return the player to the main menu. This area intention was to be the central hub of where all the gamification elements of badge unlock and progress, would be displayed to the user, giving them instant feedback on unlocked prizes and game modes, as well as hints on what to do to unlock additional features. However, due to time restrictions, this was unable to be implemented within the application. The idea was to allow the users to see how or what affects the models, from little changes to significant changes like changing the number of clusters in k-Means or the main algorithm being used to fit the clustering data. The intention was to allow the user to get hands-on with the different models, to see what they have learnt from the learning zone in motion, and also try out strategies for the game zone.

The models had three data settings, a custom one and two pre-generated datasets. The pre-generated data sets allow the user to be able to change the features that are used, assigning news features to the X and y-axis.

3.3 Evaluation of Application

To gain a deeper understanding of the application, for its effectiveness and the general overall thoughts from other peoples views, a user study got conducted. The user study involved participants in interacting with the application and then fill in an online questionnaire about what they thought of it. However, due to the coronavirus pandemic, the study got done all remotely.

The user study involved [number] of participants and got done in a quantitative style, using questionnaires of a range scale. This style of questions got decided upon due to the inability to ask the candidates follow-up questions. The participants got asked to install the application and then spend a minimum of 20 minutes interacting with it. After they had spent a minimum of 20 minutes on the application, they then needed to complete a questionnaire. The questionnaire consisted of [number of] questions with the questions either a range option style question of 1 to 5 or a short paragraph explanation. The questionnaire got conducted using Google Forms, which allowed us to have all the responses appear in a spreadsheet. Therefore, allowing reflection on the user's opinions more accessible.

[What are the results?]

Chapter 4

Implementation

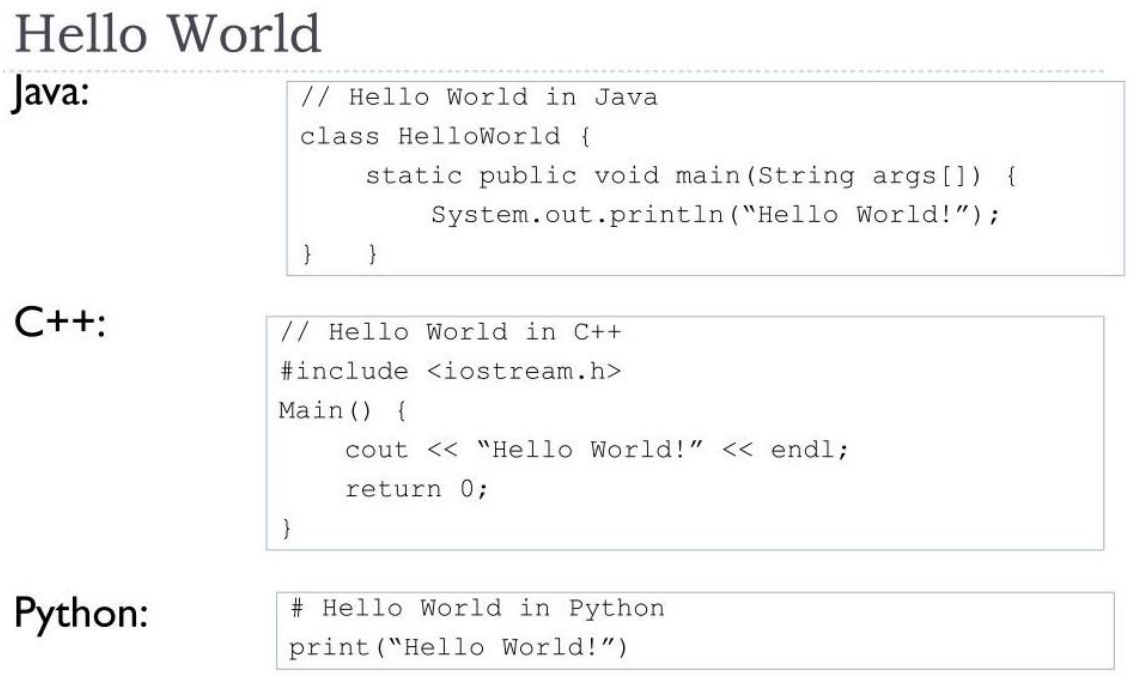
4.1 Tools

4.1.1 Programming Languages

For the implementation of our application, three primary programming languages deemed to be best suited for development. Apple's Swift programming language [25] got considered early on, due to the author's familiarisation with the programming language. The programming language gets used for creating applications for Apple's mobile and desktop operating systems, and with 1.5 billion [26] iOS devices in circulation, that was a lot of potential users. Additionally, Apple's iOS devices are prevalent within most educational settings, with Apple's iPad being one of the primary go-to devices. However, due to the language not supporting key frameworks required, or providing similar alternatives, the decision to not use this language got made.

We then got presented with three main options to use, Python, R and HTML, CSS and JavaScript.

Python is a very popular programming language [27, 28], it is fast, easy-to-use, and easy-to-deploy programming language that gets widely used to develop scalable applications. Examples include YouTube, Instagram, Pinterest and SurveyMonkey [29]. The Python Software Foundation state that Python is a high-level, object-orientated (OOP), interpreted language with dynamic semantics. Due to the language being a high-level, it has many built-in data structures. These features, along with the dynamic typing and dynamic binding together make Python attractive to development teams working in a Rapid Application Development (RAD). As Python is an extracted level above the C language [30], Python can get used as the glue that


Hello World
Java:
C++:
Python:

```
// Hello World in Java
class HelloWorld {
    static public void main(String args[]) {
        System.out.println("Hello World!");
}   }
```

```
// Hello World in C++
#include <iostream.h>
Main() {
    cout << "Hello World!" << endl;
    return 0;
}
```

```
# Hello World in Python
print("Hello World!")
```

Figure 4.1: A comparison between Java, Python and C++ to print an output to the console. [1]

connects existing components, as well as being able to be used as a scripting language [31]. Python gets considered to be easy to learn the language due to its high readability and is recommended by many exam boards as the language to use for teaching Computer Science at GCSE and A-Level level [**?**]. Python's simple and easy to learn syntax emphasises on readability, which, as a result, reduces the cost of program maintenance [31, 32].

Python gets compared to a lot of other languages. However, due to the requirements and expectations of the application, we will compare it to other similar style applications that can potentially do a similar job. These being Java, JavaScript and C++. In general, the choice of the programming language to use is many other real-world constraints, for example, financial cost, availability, training and even personal preferences and attachments. However, we will focus on language issues for the comparisons.

In comparison to Java, Python programs will typically take 3-5 times quicker (See fig: 4.1) to develop but will have a slower run time. The time difference gets attributed to Python's built-in data types and its dynamic typing [33]. As Java gets better characterised as a low-level implementation language, this would be the language of choice if application execution speed was the deciding factor. If this is not a factor, then there is no real benefit over Python.

What gets said about Java is also the same when comparing C++ to Python. It is often 5-10 times shorter than equivalent C++ code. Anecdotal evidence suggests that one Python programmer can finish in two months what two C++ programmers cannot complete in a year. Python shines as a glue language, used to combine components written in C++ [33].

In comparison to JavaScript (JS), Python's 'object-based' subset is very similar to JS. Python supports a programming method that uses simple variables and functions, similar to JS, that do not need class definitions. However, Python also supports writing for much larger programs, which leads to more reusable code by using an accurate OOP way while with JS, that is all that it can do [33].

Another language that presented itself to us was the R language. R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues [34]. Many users think of R as a statistics system [34]. Academics and statisticians have developed R over two decades. There are around 12000 packages available in CRAN (open-source repository). The wide variety of library makes R the first choice for statistical analysis, especially for specialised analytical work [35].

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering) and graphical techniques, and is highly extensible. One of R's strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed [34]. The cutting-edge difference between R and the other statistical products is the output. R has fantastic tools to communicate the results. Rstudio comes with the library knitr. Communicating the findings with a presentation or a document is easy [35].

R and Python are both open-source programming languages with a large community. New libraries or tools are added continuously to their respective catalogue. R is mainly used for statistical analysis, while Python provides a more general approach to data science. R and Python are both state of the art in terms of programming language oriented towards data science. Learning both of them is, of course, the ideal solution. R and Python requires a time-investment, and such luxury is not available for everyone. Python is a general-purpose language with a readable syntax. R, however, is built by statisticians and encompasses their specific language [35].

Python can pretty much make the same tasks as R: data wrangling, engineering, feature selection web scrapping, creating an app, for example. Python is a tool to deploy and implement machine learning at a large-scale. Python codes are easier to maintain and more robust than R. Years ago; Python did not have many data analysis and machine learning libraries.

Recently, Python is catching up and provides cutting-edge API for machine learning or Artificial Intelligence. Most of the data science job can get done with five Python libraries: Numpy, Pandas, Scipy, Scikit-learn and Seaborn [35].

Python, on the other hand, makes replicability and accessibility easier than R., if we need to use the results of our analysis in an application or website, Python is the best choice [35].

In 2019 there was an active number of 26.66 billion devices attached to the internet [36,37], with an estimation of 35 billion in 2021 [36] and by 2025 75.44 billion [37]. Experts estimate that the IoT device market will reach $1.1 trillion in 2026 [36]. Every Second 127 new devices get connected to the world wide web [36].

With so many devices on the internet, an important consideration we had was to make the application web-based. By creating the application for the internet, this would allow potentially many more people to be able to access the application and interact with the different ML models.

JS gets regarded as more of the language of the world-wide-web [38]. It got initially designed to be used client-side in a web browser. However, it has in more recent years started to branch out and be able to be used to create applications on, not only the front end of the web but also desktops, servers and mobile platforms natively. For example, React Native, Node.js and TypeScript. JavaScript is also incredibly useful, allowing developers to be able to create apps with audiences in the millions quickly [39].

The decision on what language to use we a close call between Python and JavaScript, this was due to the massive amounts of libraries that were on offer and the support communities that were in place. With both being open source and both having essential libraries available to interact with machine learning models and visualisation tools, both could have been a perfect fit for the intended application. However, we decided upon using Python. Python was chosen based on it being the go-to language for anything machine learning related, and its ability to be able to be used multiplatform on desktops or mobile devices. Python supports modules and packages, which encourages programs to be developed modularity and therefore allows code to get reused. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can get freely distributed [31]. There was also an additional factor that the author was more familiar with Python and its required libraries compared to the libraries that will get required for using JavaSript.

There was the additional decision to use HTML and CSS within a small part of the project, the 'Learning Zone', based on the quickness of being able to create and host the webpages containing the learning content. It was allowing the learning content to evolve without having any impact on the overall development of the main application, allowing the learning content to be an individual entity within the main application.

4.2 Frameworks

4.2.1 GUI Framework

With the nature of the application, we needed to make the application have a Graphical User Interface (GUI). Having a GUI allowed the learning to be a lot more hands-on and allow the players to see what is happening within the models, especially when they interact with them.

Therefore, due to the GUI requirement, three GUI libraries presented themself to us. These were Pygame, PyQT5 and Tkinter.

Pygame is a free, open-sourced library. Released under the LGPL licence, Pygame is a set of Python modules designed for writing video games. Pygame adds functionality on top of the standard Python library. Pygame allows the user to create fully featured games and multimedia programs in the python language [40].

Pygame is highly portable and runs on nearly every platform and operating system, and it gets downloaded millions of times [40].

With the main aim of the application to be a game, Pygame was a strong contender. It was providing modules that can handle a lot of the key gaming mechanics and multiple screen switching. However, it lacked some key features that were deemed essential for the application. It was unable to provide a library that could create interactable graphs to be used as data inputs for the models and be able to render HTML and CSS content for the Learning Zone. Therefore reducing the amount of flexibility, it got decided upon for using HTML and CSS for the learning content. Therefore, meaning that all the content would need to be hardcoded. If any changes were needed, a significant transformation would need to happen to the overall code, instead of just changing the web content.

PyQt is a set of Python v2 and v3 bindings for The Qt Company's Qt application framework and runs on all platforms supported by Qt including Windows, macOS, Linux, iOS and

Android. PyQt5 supports Qt v5. PyQt4 supports Qt v4 and will build against Qt v5. The bindings are implemented as a set of Python modules and contain over 1,000 classes [32].

PyQt brings together the Qt C++ cross-platform application framework and the cross-platform interpreted language Python. Qt is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets. Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create re-usable software components [32].

Qt also includes Qt Designer, a graphical user interface designer. PyQt is able to generate Python code from Qt Designer. It is also possible to add new GUI controls written in Python to Qt Designer [32].

PyQt combines all the advantages of Qt and Python. A programmer has all the power of Qt but can exploit it with the simplicity of Python [32].

Tkinter is the third option. Tkinter commonly comes bundled with Python, using Tk and is Python's standard GUI framework. It is famous for its simplicity and graphical user interface. It is open-source and available under the Python License [41].

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. It is a thin object-oriented layer on top of Tcl/Tk. Tkinter is not the only GuiProgramming toolkit for Python. It is however the most commonly used one. CameronLaird calls the yearly decision to keep TkInter "one of the minor traditions of the Python world [42]."

Tkinter supports functionality with Matplotlib, with Matplotlib offering libraries to allow handling the backend of the graph creation interacting with the GUI library. However, unlike QT, Tkinter does not support any GUI designer. Therefore the GUIs will have to be created programmatically, which will give more control, might involve more of a learning curve and potentially more time to implement in the initial stages.

After reviewing the different GUI libraries, PyQt was the decided library to use. We believed it would give us the ability to have

4.3 Packages

4.4 IDE

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development [43]. While PyCharm is a very popular IDE, and one that we have had experience with before, it is not, however, one that we have had many experiences using compared to other IDEs. While it does provide much functionality and it a lot easier to use and keep our directories organised compare to Python's provide IDE, it has, however, not been an IDE that has flowed well when we have used it.

Visual Studio Code (VS Code) is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging. First and foremost, it is an editor that gets out of the user's way. The delightfully frictionless edit-build-debug cycle means less time fiddling with the required environment, and more time executing ideas [44].

Microsoft claims that VS Code, at its heart, lightning-fast code features a lightning-fast source code editor, which is perfect for day-to-day use. With support for hundreds of languages, VS Code helps the user be instantly productive with syntax highlighting, bracket-matching, auto-indentation, box-selection, snippets, and more [44]. For serious coding, the user will often benefit from tools with more code understanding than just blocks of text. Visual Studio Code includes built-in support for IntelliSense code completion, rich semantic code understanding and navigation, and code refactoring [44]. Which we can say from experience is mostly true. However, on occasions, it has provided code completion that was not intended or needed. VS Code also allows the user to customise every feature to their liking and install any number of third-party extensions. While most scenarios work "out of the box" with no configuration, VS Code also grows with you [44]. Which, from our experience, we can say is true. VS Code has grown with us. The VS Code community has provided many extensions that have helped with our workflow.

Atom is developed and released by GitHub [45]. Atom is free, and an open-sourced code editor. Atom is a self-labelled 'a hackable text editor for the 21st century'. Atom, like VS Code, allows developers to fully customise the look, feel, and requirements to speed up their workflows.

However, Atom still allows developers to use it productively without ever touching a config file. Atom comes pre-loaded with eight syntax themes and four UI, two light and two dark, but if none of them provides any interest, Atom makes it easy and quick to install customised themes created by a third-party or to create one [45]. However, apart from pre-created extensions to help with code linting and code autocomplete abilities, none of these features is of any interest to us. The main factor does the IDE have a friendly UI and does it seem not to hinder our workflow. Which is safe to say, it does have a friendly UI and does not hinder our workflow at all.

After trailing the different IDEs, we believed that the best option going forward was the VS Code IDE. We have chosen this IDE because of two key factors. The first one being that it supported all the libraries needed, whether it was pre-installed or through downloading additional extensions, and that we have had a better familiarity with the IDE's interface from previous uses and projects.

4.5 Intricacies of the Game Components

4.5.1 Gameplay Area

[Need to Finish in application]

4.5.2 Learning Zone Area

The Learning Zone (LZ) area is an area that we intend to allow the user to do most of the learning. The LZ, in terms of UI, is very basic. It has a web browser window and three buttons. The web browser window is where the HTML and CSS documents, which the created web documents, get displayed within the application.

The web document consists of a welcome page, outlining the content, a "What is Machine Learning?", "Task Driven vs Data-Driven", "Supervised and Unsupervised Learning", "Classification", "Support Vector Machines (SVM)", "k-Nearest Neighbour", "Neural Networks", "Regression", "Linear Regression", "Logistic Regression", "Clustering", "K-Means", "Gaussian Mixture Model", "Dimensionality Reduction", "Principal Component Analysis", Linear Discriminant Analysis" and "Association Rule" web pages.

The web pages follow a similar layout design. A blue background, a yellow background layer on top with an offset grey colour behind the text. Each page contains title at the top with a dark grey background. The content of the web pages either we an overview, for example, "Clustering", which looked into clustering as a whole and what the different types were. Alternatively, a web page would explain a specific algorithm, for example, "K-Means", which explained the intricacies of how the algorithm worked and the critical mechanics behind it.

The three buttons at the bottom of the application screen trigger three different actions. All of which match to the intended buttons, a home button, to go back to the main menu, a free play button to send the player to the free play area with the intended algorithm that the user was learning about, and a quiz button which loads a multi-choice quiz.

When the player clicks the free play button, the application checks the HTML documents title tag and loads up the required model in the free play section. However, if the player clicks the button and the web page does not have a model available, within the free play section or it is just a general overview page, a message box will appear. The message box intension is to let the user know that they can not progress to the free play zone and a list of the available models (see fig: **??**).

The Quiz area is an additional area to the learning zone. The Quiz area is where the user can get tested on what they learned in the LZ area, in the form of a multiple-choice quiz. When the user is viewing a topic on the LZ, and they decide to take a quiz, the user will click on the quiz button, and this will read the title tags of the HTML document and open the required quiz. The quiz questions and answers are within their text file, and the name of the file matches the title tag's content. The text file itself holds the information in the format of a 2D array, that has the question at position zero, the answer at position one and then position 2 to 5 are the multiple-choice options. This information from the text file populates a question label and four buttons, allowing the user to click what button they think is the answer. The total of correct answers get added up and displayed to the user at the end in a message box.

4.5.3 Free Play Area

The Free Play (FP) zone is an area where the user gets to interact and play with different ML models. The models include Linear Regression, K-Means, Neural Networks, Linear Discriminant Analysis, Gaussian Mixture Models and SVM. The intension for the FP zone was to have all the models explained in the learning zone be available for the player to interact with, so it could help them fully understand how the model works by allowing the user to manipulate parameters and data points. However, due to time restrictions, there are only six models available, with 5 of the models having real interactivity but to different degrees.

When the FP zone is accessed, unless accessed through the Learning Zone, a randomly selected model gets displayed to the user from the list mentioned before. On first glance, the user has multiple areas to either interact with or present information to them. The screen has a Widget that is linked to a Matplotlib library to handle PyQT5 backend interactions. Also, a model overview is displayed next to the widget, it tells the user information about the model, for example, the type of learning it is, supervised or unsupervised, the name of the model and a brief overview of the model. Just beneath the widget and overview is a group box that contains all the settings for the model and data interaction. The model settings group box contains combo boxes, radio buttons, checkboxes, line edits and buttons, which all do different things depending on the model and data sets selected. Within the model settings group box, there are three additional group boxes. These are 'Model Attribute(s), 'Model Parameter(s)' and 'Data Options' with each group box displaying different content depending upon the model and data options selected in the combo boxes.

The model combo box contains six values, and these are 'Please Select', 'K-Means', 'LDA', 'Linear Regression', 'GMM', 'SVM', 'Neural Networks'. Once one of these options is selected, apart from the 'Please Select', the desired model will display in the Matplotlib widget area. The Model attributes and parameters boxes will display the required information unless the models 'LDA', 'SVM' and 'GMM' are selected. Instead, a label placeholder saying, 'No Options available, yet!' will be displayed. While LDA has a fully interactive model in the MatplotLib widget, it does not present options for the user to change within the model, the user can only click on the widget and place points, which the model will then apply and create the required actions. Therefore a place holder label appears stating to click in the game widget to interact with the model. While LDA and GMM both have the ability for the model visualisations to toggle on and off, showing how the models have fit their data, GMM has little much additional functionality. GMM only allows the user to toggle on and off the visualisation, which is the model predicting the 'Iris' dataset clusters. However, SVM only displays the model's output, again using the 'Iris' data set, but the output shows the boundary lines and area that each partition covers.

While on the other hand, the Linear Regression, K-Means and Neural Network models display different options. Linear Regression displays to the user labels in the attributes group box to show them the values for the intercept, estimated coefficient and outcome. There is also a line edit available for the user to input a value and see what the model would predict out, which gets displayed in the output label. However, Linear Regression does not have any model parameters, and this is due to the values getting deemed as not having much impact on the model and limited implementational time.

When K-Means gets selected, both the attributes and parameters group boxes have information and selection options displayed to the user. The attributes group box displays the information for Inertia, the number of iterations that got performed fitting the data, prediction, which relies on the user to click within the Matplotlib widget and the X and y coordinates get displayed along with a cluster prediction label in the output label. There is also a distance from the centroid value displayed, and this value got achieved by using the SKLearn Metrics library. The model parameters group box displays multiple line edits and a combo box that allows the user to input values to the model. These will alter the K-Means k value (number of clusters), the number of initialisers, the max number of iterations and the underlying algorithm (auto, full or Elkan), that gets used. The k value is independent of the number of clusters in the data options, so they do not impact on each other. Allowing the k value to be changed independently will allow the user to be able to experiment with the model to see how two, three or other k values affect the prediction, even when known that the data may have, for example, five different clusters. K-Means also brings up an additional option, and this is to be able to switch on and off the centres of the clusters. When the checkbox gets enabled, this will lay on top of the data points an 'X' where each of the cluster centres is and when it is disabled, it will remove the 'X'.

[NN Att no. of layers/ neurons -> params set the values. Not implemented in-app yet!]

The data combo box displays the data options for the different models and depending on what option is selected depends on the information that is on offer to the user in the data options group box. If the custom data option is selected, then the group box will display different options to the user to be able to generate custom data points to be displayed in the Matplotlib widget game screen. The only models that have this option are the Linear Regression and K-Means models. Linear Regression has the data options 'Diabetes' and 'Boston House Prices' and K-Means has the data options 'Iris' and 'Moons'. When these are selected, the data option displays radio button options for the user to select the features they would like the model to display, on the X and y-axis, and get fitted. Linear Regression's 'Custom' data option allows the user the ability to change the random generated data's settings. These settings include the number of data samples and if there are any outliers wanted, if so then an option to add the number of outliers. Whereas K-Means 'Custom' data option allows the user the ability to change the number of clusters to generate and the number of data samples wanted. The other models have a label placeholder saying, 'no data options selected yet!' and no actual data selection options. In the case of LDA and Neural Network, this is more due to the decision we made. Based on the way the model's fit function was implemented, not allowing the user to generate random data was decided. Doing so would impact on how the model gets interacted with by the user. However, in the case of the other models, it was a lack of time that impacted the inability to add this feature. Though, it was always the intention to add it.

The final aspects of the Model options group box are three buttons, a play button, a clear button and a home button. Where the home button is self-explanatory in terms of returning the user to the main menu, and the clear button resetting the Matplotlib Widget axis contents. The play button is where the primary handling of how the user interacts with the back end of the models and datasets. When the user inputs information, they are required to press the play button for these features to be implemented.

4.5.4 Achievements Area

The intensions for the Achievement Area was displaying to the user all of the gamification badges available. Providing an overview and hits on how to unlock them. The achievements were going to have a bronze, silver and gold level, and we planned to be unlocked once the user had completed specific tasks like playing the game three times or completing a quiz. However, due to time limitation, this was not possible, and the application, as it currently stands, displays a coming soon image and button for the user to navigate back to the main menu.

4.6 Example user stories (A UML term for case studies or example playthroughs)

Chapter 5

Conclusions and Future Work

In this document we have demonstrated the use of a LaTeX thesis template which can produce a professional looking academic document.

5.1 Contributions

The main contributions of this work can be summarized as follows:

- **A LaTeX thesis template**

  Modify this document by adding additional top level content chapters. These descriptions should take a more retrospective tone as you include summary of performance or viability.

- **A typesetting guide of useful primitive elements**

  Use the building blocks within this template to typeset each part of your document. Aim to use simple and reusable elements to keep your document neat and consistently styled throughout.

- **A review of how to find and cite external resources**

  We review techniques and resources for finding and properly citing resources from the prior academic literature and from online resources.

5.2 Future Work

Future editions of this template may include additional references to Futurama.

Bibliography

[1] Things Tech, "Which programming language to start with as a beginner," 2020, [Online; accessed August 10th, 2020]. [Online]. Available: https://thingsteck.wordpress.com/

[2] Quora, "What are the top 5 misconceptions surrounding artificial intelligence and machine learning?" Retrieved May 10, 2020 from: https://www.quora.com/What-are-the-top-5-misconceptions-surrounding-Artificial-Intelligence-and-Machine-Learning.

[3] MathWorks, "Introducing machine learning."

[4] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[6] S. Walz, in *The Gameful World: Approaches, Issues, Applications*. MIT Press, 2015.

[7] Growth Engineering. (2019) The history of gamification: From the very beginning to right now. [Online]. Available: https://www.growthengineering.co.uk/history-of-gamification/

[8] Wikipedia. (2020) Gamification. [Online]. Available: https://en.wikipedia.org/wiki/Gamification

[9] J. McGonigal. (2010) Gaming can make a better world. [Online]. Available: https://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world?language=en#t-1184578

[10] Gamification: Using Game-Design Elements in Non-Gaming Contexts. (2011) CHI. [Online]. Available: http://chi2011.org/communities/games/index.html

[11] Wikipedia. (2020) Data aggregation. [Online]. Available: https://en.wikipedia.org/wiki/Data_aggregation

[12] R. Swatman. (2016) Pokémon go catches five new world records. [Online]. Available: https://www.guinnessworldrecords.com/news/2016/8/pokemon-go-catches-five-world-records-439327

[13] R. Silverman, "Latest game theory: Mixing work and play — companies adopt gaming techniques to motivate employees." Wallstreet Journal, 2011.

[14] M. Herger. (2012) Gamification and law or how to stay out of prison despite gamification. [Online]. Available: https://web.archive.org/web/20120425121358/http:/enterprise-gamification.com/index.php/en/blog/4-blog/65-gamification-and-law-or-how-to-stay-out-of-prison-despite-gamification

[15] Wikipedia. (2020) Gamification of learning. [Online]. Available: https://en.wikipedia.org/wiki/Gamification_of_learning

[16] A. Deese. (2020) 5 benefits of gamification. [Online]. Available: https://ssec.si.edu/stemvisions-blog/5-benefits-gamification

[17] H. Deese. (2012) Using gamification to aid in adolescent development in the classroom: Cognitive and physical processes can enhance growth. [Online]. Available: http://www.ashleydeese.com/2012/09/26/using-gamification-to-aid-in-adolescent-development-in-the-classroom-cognitive-and-physical-processes-can-enh

[18] A. Blum-Dimaya, S. A. Reeve, K. F. Reeve, and H. Hoch, "Teaching children with autism to play a video game using activity schedules and game-embedded simultaneous video

modeling," in *Education and Treatment of Children*. West Virginia University Press, 2010, pp. 351–370.

[19] M. Doyle. (2014) Learning with simcity: Valuable lessons kids can learn playing mayor. [Online]. Available: https://www.brightpips.com/learning-simcity-valuable-lessons-kids-learn-playing-mayor/

[20] Wikipedia. (2020) Serious game. [Online]. Available: https://en.wikipedia.org/wiki/Serious_game

[21] M. Baaden, O. Delalande, N. Ferey, S. Pasquali, J. Waldispühl, and A. Taly, "Ten simple rules to create a serious game, illustrated with examples from structural biology." Public Library of Science, 2018.

[22] R. Follett and V. Strezov, "An analysis of citizen science based research: usage and publication patterns," *PloS one*, vol. 10, no. 11, p. e0143687, 2015.

[23] L. Mazzanti, S. Doutreligne, C. Gageat, P. Derreumaux, A. Taly, M. Baaden, and S. Pasquali, "What can human-guided simulations bring to rna folding?" *Biophysical journal*, vol. 113, no. 2, pp. 302–312, 2017.

[24] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[25] Apple Inc. (2020) Swift. [Online]. Available: https://developer.apple.com/documentation/swift

[26] M. Potuck. (2020) Apple hits 1.5 billion active devices with 80iphones and ipads running ios 13. [Online]. Available: https://9to5mac.com/2020/01/28/apple-hits-1-5-billion-active-devices-with-80-of-recent-iphones-and-ipads-running-ios-13/

[27] K. Finley. (2020) Python is more popular than ever. [Online]. Available: https://www.wired.com/story/python-language-more-popular-than-ever/

[28] B. Popper. (2020) The 2020 developer survey results are here! [Online]. Available: https://stackoverflow.blog/2020/05/27/2020-stack-overflow-developer-survey-results/

[29] A. Goel. (2020) Best programming language to learn in 2020 (for job and future). [Online]. Available: https://hackr.io/blog/best-programming-languages-to-learn-2020-jobs-future

[30] Stack Overflow. (2013) Python vs cpythony. [Online]. Available: https://stackoverflow.com/questions/17130975/python-vs-cpython

[31] Python Software Foundation. (2020) What is python? executive summary. [Online]. Available: https://www.python.org/doc/essays/blurb/

[32] Riverbank Computing. (2020) What is pyqt? [Online]. Available: https://riverbankcomputing.com/software/pyqt/intro

[33] Python Software Foundation. (2020) Comparing python to other languages. [Online]. Available: https://www.python.org/doc/essays/comparisons/

[34] The R Foundation. (2020) What is r? [Online]. Available: https://www.r-project.org/about.html

[35] Guru 99. (2020) R vs python: What's the difference? [Online]. Available: https://www.guru99.com/r-vs-python.html#:~:text=New%20libraries%20or%20tools%20are,general%20approach%20to%20data%20science.&text=Python%20is%20a%20general%2Dpurpose,and%20encompasses%20their%20specific%20language.

[36] G. D. Maayan. (2020) The iot rundown for 2020: Stats, risks, and solutions. [Online]. Available: https://securitytoday.com/Articles/2020/01/13/The-IoT-Rundown-for-2020.aspx?Page=1

[37] Statista. (2020) Internet of things (iot) connected devices installed base worldwide from 2015 to 2025. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[38] World Wide Web Foundation. (2020) History of the web. [Online]. Available: https://webfoundation.org/about/vision/history-of-the-web/

[39] T. DeGroat. (2019) The history of javascript: Everything you need to know. [Online]. Available: https://www.springboard.com/blog/history-of-javascript/

[40] Pygame. (2020) About. [Online]. Available: https://www.pygame.org/wiki/about

[41] A. Sharma. (2020) Introduction to gui with tkinter in python. [Online]. Available: https://www.datacamp.com/community/tutorials/gui-tkinter-python?utm_source=adwords_ppc&utm_campaignid=

898687156&utm_adgroupid=48947256715&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=229765585186&utm_targetid=aud-299261629574:dsa-429603003980&utm_loc_interest_ms=&utm_loc_physical_ms=1007460&gclid=Cj0KCQjwsuP5BRCoARIsAPtX_wGxaCEuqKDVDDyVASqiCw2zIKYwN2Duo3DObWGcfwQAjH3oxWK5WfoaAhafEALw_wcB

[42] Python Software Foundation. (2020) Tkinter. [Online]. Available: https://wiki.python.org/moin/TkInter

[43] PyCharm. (2020) Get started. [Online]. Available: https://www.jetbrains.com/help/pycharm/quick-start-guide.html

[44] Microsoft. (2020) Why did we build visual studio code? [Online]. Available: https://code.visualstudio.com/docs/editor/whyvscode

[45] CloudApp. (2020) A guide to atom text editor. [Online]. Available: https://www.getcloudapp.com/blog/how-to-use-atom-text-editor

Appendix A

Implementation of a Relevant Algorithm

```c
#include <stdio.h>

int main(int argc, char *argv[]) {
  printf("Hello world.\n");
  return 0;
}
```

Listing A.1: An implementation of an important algorithm from our work.

Appendix B

Supplementary Data

The results of large ablative studies can often take up a lot of space, even with neat visualization and formatting. Consider putting full results in an appendix chapter and showing excerpts of interesting results in your chapters with detailed analysis. You can use labels and references to refer the reader here for the full data.

```
from scipy.ndimage import sobel
from tensorflow.keras.callbacks  import EarlyStopping
from tensorflow.keras            import backend as K
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers     import Input, Dense, Lambda, Concatenate
from tensorflow.keras.models     import Model
import tensorflow as tf
from sklearn.metrics import confusion_matrix
from random import randint
import matplotlib.pyplot as plt
import numpy as np

from PyQt5.QtWidgets import *
from matplotlib.backends.backend_qt5agg import FigureCanvas
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg
from matplotlib.figure import Figure
import matplotlib

import os

import traceback

import numpy as np

matplotlib.use('Qt5Agg')

```

```python

class NNGameboard(QWidget):
    model_name = "Neural Network (ANN)"
    learning_type = "Supervised Learning"
    model_overview = "A neural network has input and output neurons, which are connected by weighted synapses.\n\n" + \
                     "The weights affect how much of the forward propagation goes through the neural network.\n\n The weights can then be changed during the back propagation " + \
                     "     this is the part where the neural network is now learning.\n\nThis process of forward propagation and backward propagation is conducted iteratively on every piece of data " + \
                     "in a training data set.\n\n" +\
                     "Click in graph widget to start!"


    def __init__(self, parent=None, game_mode=""):
        QWidget.__init__(self, parent)

        self.model = self.make_model()
        self.model.compile(Adam(), loss='binary_crossentropy', metrics=['binary_accuracy'])

        self.num_players = 2
        self.players     = [[], []]

        self.game_round  = 0
        self.turn        = 0
        self.game_player = 0


        self.retrain    = False
        ### from LDA
        self.pointOwner = []
        self.playerID   = False
        self.points     = []

        self.res         = 100
        self.xs          = np.linspace(-1, 1, self.res)
        self.ys          = np.linspace(-1, 1, self.res)
        self.xv, self.yv = np.meshgrid(self.xs, self.ys)
        self.xv          = np.reshape(self.xv, (-1,))
        self.yv          = np.reshape(self.yv, (-1,))
        self.grid        = np.stack([self.xv, self.yv], axis=-1)
        self.grid_preds  = np.ones((self.res, self.res)) * 0.5
        self.loss, self.acc = 0, [0]
        self.player_colours = ['b', 'r']#np.array([self.player_colours(64), self.
            player_colours(255-64)])

        # Canvas setup
        self.canvas    = FigureCanvas(Figure())
        self.fig       = self.canvas.figure
        self.canvas.ax = self.fig.add_subplot(111)
        self.cid       = self.canvas.figure.canvas.mpl_connect('button_press_event
            ', self)

        self.canvas.ax.set_xlim([-1, 1])
        self.canvas.ax.set_ylim([-1, 1])

        self.vertical_layout = QVBoxLayout()
        self.vertical_layout.addWidget(self.canvas)

        self.setLayout(self.vertical_layout)

    def make_model(self):
        # Fully connected NN #Dense means Fully Connected
        inputs   = Input(shape=(2,))

        f_sin    = Lambda(lambda x: K.sin(x))(inputs)
        f_sq     = Lambda(lambda x: K.square(x))(inputs)
        f_corr   = Lambda(lambda x: K.prod(x, axis=-1, keepdims=True))(inputs)
        features = Concatenate()([inputs, f_sin, f_sq, f_corr])

        x        = Dense(10, activation='relu')(features)  # input
        outputs  = Dense(1,  activation='sigmoid')(x)

        model    = Model(inputs=inputs, outputs=outputs)

        self.no_layers         = 2
        self.no_neurons        = 10
        self.l0_activation     = 'Relu'
        self.output_activation = 'Sigmoid'

        return model


    def sample_normal(self, shape, mu, sig):
        return (np.random.normal(size=shape) * sig) + mu


    def reset_weights(self, model):
        session = K.clear_session()


    def __call__(self, event):
        try:
            num_points_per_round = 2
            samples_per_point    = 100
            variance             = 0.1

            if event.inaxes != self.canvas.ax:
                return

            self.ix, self.iy = event.xdata, event.ydata
            new_point        = np.array([self.ix, self.iy])
            self.players[self.game_player] += [new_point]

            ## from LDA

            self.points.append([self.ix, self.iy])
            self.pointOwner.append(self.playerID)

            if self.playerID == True:
                self.canvas.ax.scatter(self.ix, self.iy,
                                       s=100, c=self.player_colours[1], edgecolors
                                           ='w')
            else:
                self.canvas.ax.scatter(self.ix, self.iy,
                                       s=100, c=self.player_colours[0], edgecolors
                                           ='w')

            self.playerID = not self.playerID

            if (self.retrain):
                self.game_round += 1
                sample_points    = []
                sample_labels    = []

                for player_id in range(self.num_players):
                    for point in self.players[player_id]:
                        sample_points += [self.sample_normal((samples_per_point,
                            2), point, np.array([[variance, variance]]))]
                    sample_labels += [np.ones((len(self.players[player_id])*
                        samples_per_point,), dtype=np.int32) * player_id]

                sample_points = np.concatenate(sample_points, axis=0)
                sample_labels = np.concatenate(sample_labels, axis=0)

                self.reset_weights(self.model)

            halting = EarlyStopping(monitor='binary_accuracy', min_delta=0, patience=5, verbose=0, mode='max')
            h       = self.model.fit(x=sample_points, y=sample_labels,
                                     batch_size=32, epochs=100,
                                     callbacks=[halting])

            self.loss, self.acc = h.history['loss'][-1], h.history['binary_accuracy']

            self.grid_preds = self.model.predict(self.grid)
            self.grid_preds = np.reshape(self.grid_preds, (self.res, self.res))

            sample_points   = []
            sample_labels   = []

            for player_id in range(self.num_players):
                for point in self.players[player_id]:
                    sample_points += [point]

                sample_labels += [player_id] * len(self.players[player_id])

            sample_points = np.stack(sample_points, axis=0)
            sample_labels = np.array(sample_labels, dtype=np.int32)

            sample_preds  = self.model.predict(sample_points)
            sample_preds  = (sample_preds > 0.5).astype(np.int32)


        self.a1 = np.sum(self.grid_preds >= 0.5) / (self.res**2)
        self.a0 = 1. - self.a1

        self.territory = 'Territory {} | {}'.format((round(self.a0, 2) * 100), (round(self.a1, 2) * 100))
        #print('Territory {} | {}'.format(self.a0, self.a1))
        #print(self.territory)

        self.turn  += 1
        self.game_player = self.turn % self.num_players

        self.retrain = self.turn > (num_points_per_round-1) and ((self.turn-1) %
                                                            (num_points_per_round * self.num_players)) == 0

            self.canvas.ax.imshow((self.grid_preds-0.5)*0.6, extent=(-1, 1, 1, -1)
                , vmin=-0.5, vmax=0.5,
                                  interpolation='bilinear', cmap="cool" )#self.
                                      player_colours[self.game_player])

            self.fig.canvas.draw()

        except Exception:
            print("In exception")
        #    self.canvas.ax.set_title(traceback.format_exc())

    def clear_values(self):
        pass
```

Listing B.1: The NN Code